# On Simulating Nondeterministic Stochastic Activity Networks[1]

Valmir C. Barbosa[2] Fernando M. L. Ferreira[3] Daniel V. Kling[4]

Eduardo Lopes[5] Fábio Protti[6] Eber A. Schmitz[7]

**Abstract.** In this work we deal with a mechanism for process simulation called a *NonDeterministic Stochastic Activity Network* (NDSAN). An NDSAN consists basically of a set of *activities* along with *precedence relations* involving these activities, which determine their order of execution. Activity durations are stochastic, given by continuous, nonnegative random variables. The nondeterministic behavior of an NDSAN is based on two additional possibilities: (i) by associating choice probabilities with groups of activities, some branches of execution may not be taken; (ii) by allowing iterated executions of groups of activities according to predetermined probabilities, the number of times an activity must be executed is not determined *a priori*. These properties lead to a rich variety of activity networks, capable of modeling many real situations in process engineering, project design, and troubleshooting. We describe a recursive simulation algorithm for NDSANs, whose repeated execution produces a close approximation to the probability distribution of the completion time of the entire network. We also report on real-world case studies.

**Keywords:** activity networks, stochastic activity networks, nondeterministic activity networks, stochastic project scheduling problems.

## 1 Introduction

In this work we deal with a mechanism for process simulation called a *NonDeterministic Stochastic Activity Network* (NDSAN). An NDSAN consists basically of a set of *activities* along with *precedence relations* involving these activities, which determine their order of execution. This order is captured by a digraph with some special properties: the possibility of defining *nondeterministic branches of execution*, by associating choice probabilities with some activities, and *loops of execution*, which specify the iterated execution of a group of activities according to predetermined loop

[1]This work is partially supported by CNPq, CAPES, and a FAPERJ BBP grant.
[2]COPPE/PESC, UFRJ, C.Postal 68511, Rio de Janeiro, Brazil, CEP 21941-972. E-mail: valmir@cos.ufrj.br
[3]NCE, UFRJ, C.Postal 2324, Rio de Janeiro, Brazil, CEP 20001-970. E-mail: fmachado@nce.ufrj.br
[4]NCE, UFRJ, C.Postal 2324, Rio de Janeiro, Brazil, CEP 20001-970. E-mail: danielvk@posgrad.nce.ufrj.br
[5]NCE, UFRJ, C.Postal 2324, Rio de Janeiro, Brazil, CEP 20001-970. E-mail: eduardolopes@gmx.net
[6]IM and NCE, UFRJ, C.Postal 2324, Rio de Janeiro, Brazil, CEP 20001-970. E-mail: fabiop@nce.ufrj.br
[7]IM and NCE, UFRJ, C.Postal 2324, Rio de Janeiro, Brazil, CEP 20001-970. E-mail: eber@nce.ufrj.br

probabilities. These properties allow for a rich variety of activity networks, capable of modeling many real situations in process engineering, project design, and troubleshooting.

There are two main types of activity networks. A *deterministic activity network* is represented by a precedence digraph whose topology remains *fixed* as the activities are executed. Examples of deterministic activity networks include CPM and PERT networks, see e.g. [10]. On the other hand, a nondeterministic activity network allows for the possibility of a dynamic topology. Examples of such networks are inhomogeneous Markov chains, GANs (Generalized Activity Networks) [4], and GERT (Graphical Evaluation and Review Technique) networks [11].

The duration of each network activity is given by a random variable. Thus, a fundamental problem is determining the distribution of the completion time of the entire network. For deterministic activity networks, this general problem is known as the Stochastic Project Scheduling Problem [3].

Our definition of NDSANs combines stochastic activity durations with nondeterminism. In an NDSAN, activities are represented by *nodes*, and an arc oriented from activity $a_i$ to activity $a_j$ means that the execution of $a_j$ may only start after the execution of $a_i$ has ended. Nondeterminism is achieved, as indicated above, by means of two possibilities: (i) some branches of execution are not necessarily taken, and (ii) the number of times a group of activities is to be executed is not determined *a priori*. These additional possibilities are supported by the introduction of two new categories of nodes, namely *decision nodes* and *loop nodes*. A decision node associates *probabilities* with its out-neighbors and selects one of them to be executed accordingly; this selection is interpreted as one possible deterministic scenario among many. A loop node allows the repeated execution of a group of activities, the number of iterations depending on probabilities associated with the loop node. Loop nodes are particularly interesting to model refinement processes, such as quality control and error testing/correction. We also define *junction nodes* for adequately combining the two new constructions into the network. In Section 2 we define NDSANs formally, in terms of recursive construction steps that combine smaller NDSANs into larger ones via certain types of structured templates.

In Section 3, we give an analytical description of the random variable $T[D]$ associated with the completion time of NDSAN $D$. We assume that the duration of each activity $a_i$ in $D$ is given by a continuous, nonnegative random variable $T_i$. The random variable $T[D]$ is thus given in terms of the $T_i$'s and the probabilities associated with the decision/loop nodes.

Although $T[D]$ can be described precisely, we lack a closed-form expression for it and even numerical methods to find its distribution from such a description may be computationally too hard, especially when the number of activities is large. In Section 4, we describe a recursive simulation algorithm whose execution returns a single plausible value ("observation") in the sample space of $T[D]$. Running the simulation algorithm a suitable number $N$ of times produces a close approximation to the probability distribution of $T[D]$. The value of $N$ can be obtained by using the same statistic as the Kolmogorov-Smirnov test, see e.g. [7] (Section 13.5), as we also discuss in Section 4.

Section 5 presents two computational experiments. For each experiment, the result of the simulations is shown as a frequency histogram together with a fitting curve that approximates the expected shape of the density of $T[D]$, an approximate probability distribution of $T[D]$, and an approximate probability density of $T[D]$ obtained from the approximate distribution. Section 6 discusses ongoing work.

In a recent related work, Leemis *et al.* [9] develop algorithms to calculate the probability distribution of the completion time of a stochastic activity network with continuous activity durations. In their work, activities are modeled by *arcs* and the networks are acyclic and deterministic (i.e., allow no variation in topology). The authors describe a recursive Monte Carlo simulation algorithm, which is network-specific and must therefore be rewritten specifically for each new network. Also, they provide two exact algorithms, one for series-parallel networks and another for more general networks whose nodes have at most two incoming arcs each.

We remark that all the discussion on random variables in this work can be adapted to the case of discrete random variables. (In [13], pp. 122–123, for example, an activity network with discrete activity durations is given.)

## 2 Formal definition of NDSANs

In this work, $D$ denotes a digraph with $n$ nodes and $m$ arcs. If $(v, w)$ is an arc of $D$, then node $v$ is an *in-neighbor* of node $w$, whereas $w$ is an *out-neighbor* of $v$. By disregarding arc orientation, we may also simply say that $v$ and $w$ are *neighbors*. A node having no in-neighbors (resp. out-neighbors) is called a *source node* (resp. *sink node*). If $D$ is a digraph containing a single source (resp. sink) node $v$ , then $v$ is denoted by $source(D)$ (resp. $sink(D)$).

An NDSAN is a special digraph whose node set is partitioned into four subsets of nodes: a subset $S_a = \{a_i \mid 1 \le i \le n_a\}$ of *activity nodes*; a subset $S_b = \{b_i \mid 1 \le i \le n_b\}$ of *junction nodes*; a subset $S_d = \{d_i \mid 1 \le i \le n_d\}$ of *decision nodes*; and a subset $S_\ell = \{\ell_i \mid 1 \le i \le n_\ell\}$ of *loop nodes*.

An *activity node* $a_i$ represents a single *activity* (or *task*) to be executed in the network. The execution of $a_i$ starts only after the execution of *all* of its in-neighbors has ended. When the execution of $a_i$ ends, *all* of its out-neighbors start executing simultaneously. Each activity node $a_i$ has a *duration* (*execution time*) $T_i$, which is a continuous, nonnegative random variable. We assume that the execution time of an activity node does not depend on the execution time of any other activity node. That is, the $T_i$'s are independent random variables. An activity node is represented by a circle. See Figure 1(a).

A *junction node* $b_i$ is used for a syntactic purpose. It may have several in-neighbors, but it has a single out-neighbor $v$. When the execution of any in-neighbor of $b_i$ ends, the execution of $v$ is started immediately. In other words, $b_i$ acts simply as a "connecting point" of incoming arcs. A junction node is represented by a square. See Figure 1(b).

A *decision node* $d_i$ is used to select one particular branch of the execution flow, as described in what follows. By construction, all of $d_i$'s neighbors are activity nodes. It has a single in-neighbor $a_h$ and $\alpha_i \ge 2$ out-neighbors $a_{j_1}, \ldots, a_{j_{\alpha_i}}$. The execution of $d_i$ is assumed to be instantaneous, and consists of selecting exactly one of its out-neighbors, say $a_{j_k}$, as the next node to execute. The activity node $a_{j_k}$ is selected by $d_i$ with probability $p^i_k$, $k = 1, \ldots, \alpha_i$, such that $\sum_{k=1}^{\alpha_i} p^i_k = 1$. A decision node is represented by a lozenge. See Figure 1(c).

A *loop node* $\ell_i$ represents the usual iteration mechanism. By construction, $\ell_i$ has a single in-neighbor (a junction node $b_h$) and two out-neighbors (activity nodes $a_r$ and $a_j$). After the execution of $b_h$, a Boolean condition $E_i$ associated with $\ell_i$ is instantaneously tested: if $E_i$ is *false* then $a_r$ is executed next, otherwise $a_j$ is. An array of real values associated with $\ell_i$ gives the sequence $q^i_1, \ldots, q^i_{\beta_i}$ of probabilities corresponding to $\beta_i$ consecutive passages through $\ell_i$, in such a way that the probability that $E_i$ is *false* at the $k$th passage through $\ell_i$ is $q^i_k$. That is, the probability of exiting the loop at this point is $1 - q^i_k$. We assume that $q^i_{\beta_i} = 0$ in order to guarantee the termination of the loop in at most $\beta_i$ consecutive passages through $\ell_i$. A loop node is represented by a filled lozenge. See Figure 1(d).

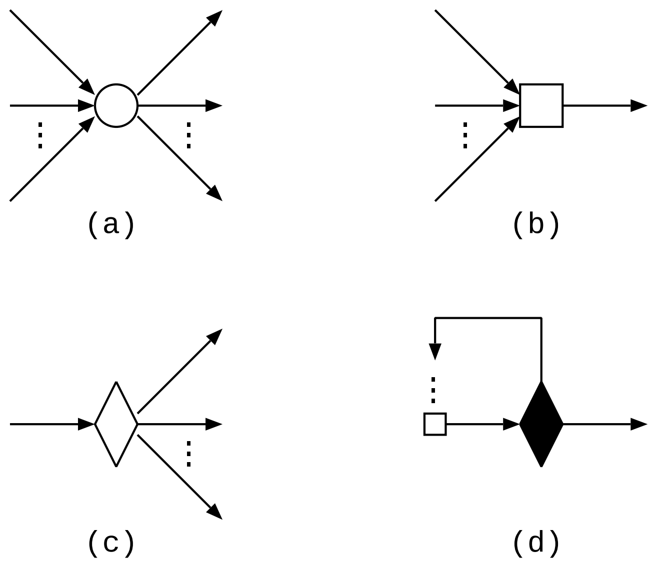


Figure 1: Types of node: (a) activity node; (b) junction node; (c) decision node; (d) loop node.

We are now ready to give the formal definition of NDSANs in terms of recursive construction steps. The base NDSAN is a digraph consisting of a single activity node. In a general step, NDSANs containing a single source node and a single sink node are combined to yield a larger NDSAN.

The recursive construction steps are based on the following *Substitution Rule:*

**Substitution Rule:** Let $D_0$ be a digraph and $\{v_1, v_2, \ldots, v_\eta\}$ a subset of its node set. Let $D_1, D_2, \ldots D_\eta$ be NDSANs, each containing a single source node and a single sink node. Construct an NDSAN $D$ by replacing $v_i$ by $D_i$, $1 \leq i \leq \eta$, in such a way that every input (output) arc of $v_i$ in $D_0$ is an input (output) arc of $source(D_i)$ ($sink(D_i)$) in $D$. Let $Sub(D_0, D_1, \ldots, D_\eta) = D$.

**Definition 1** *An NDSAN is defined as follows:*

1. *A digraph $D$ consisting of a single activity node is an NDSAN, called the* **trivial NDSAN**.

2. *Let $D_1, D_2, \ldots D_\eta$ be NDSANs.*

   2.1 *If $D_0$ is an acyclic digraph of node set $\{v_1, \ldots, v_\eta\}$ containing a single source node and a single sink node (Figure 2(a)), then $Sub(D_0, D_1, \ldots, D_\eta)$ is an NDSAN, called an* **acyclic NDSAN** *(Figure 2(b)).*

   2.2 *If $D_0$ is the digraph in Figure* 3(a)*, then $Sub(D_0, D_1, \ldots, D_\eta)$ is an NDSAN, called a* **decision NDSAN** *(Figure 3(b)).*

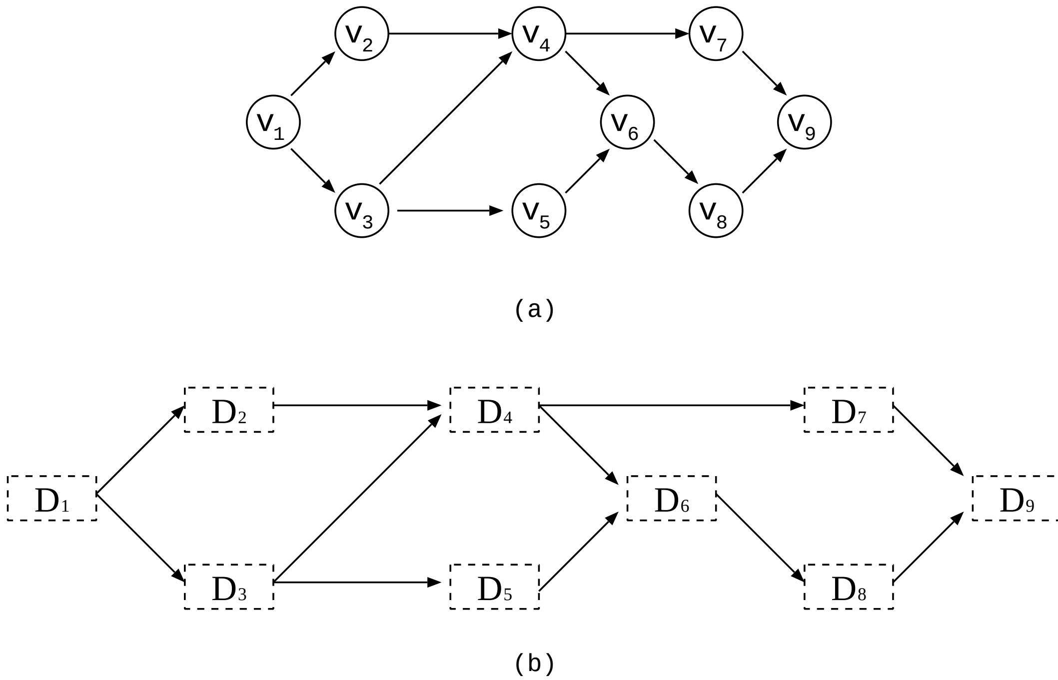


Figure 2: Construction of an acyclic NDSAN.

2.3 *If $D_0$ is the digraph in Figure* 4(a)*, then $Sub(D_0, D_1, D_2, D_3)$ is an NDSAN, called a* **loop NDSAN** *(Figure 4(b)).*

It is easy to see that the network $Sub(D_0, D_1, \ldots, D_\eta)$ resulting from 2.1, 2.2, or 2.3 in the above definition contains a single source node and a single sink node, both activity nodes.

**Scope of the definition of NDSANs.** Although other definitions of NDSANs may be possible, we believe that Definition 1 not only determines a wide class of activity networks, but also allows the realization of any structured project, since it provides basic constructions that are generally thought to suffice for the specification of how concurrent tasks are to interrelate. In other words:

- an acyclic NDSAN embodies the notion of multiple concurrent execution threads, which may be started as a single thread branches out into several independent ones, and terminated as they coalesce into a single thread for further execution.
- a decision NDSAN allows for nondeterministic switches, or decision points, to be incorporated into the course of a thread's execution.
- a loop NDSAN allows any of the above to be iterated, possibly for a probabilistically selected number of times.

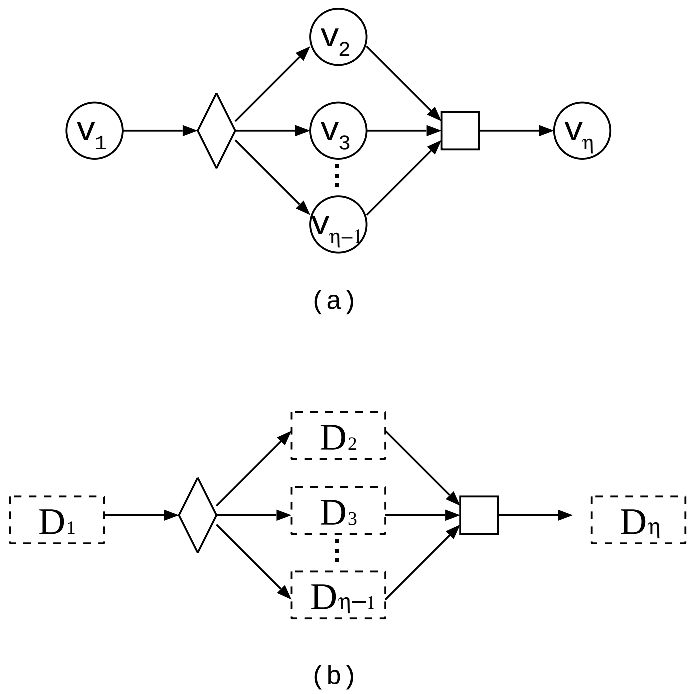


Figure 3: Construction of a decision NDSAN.

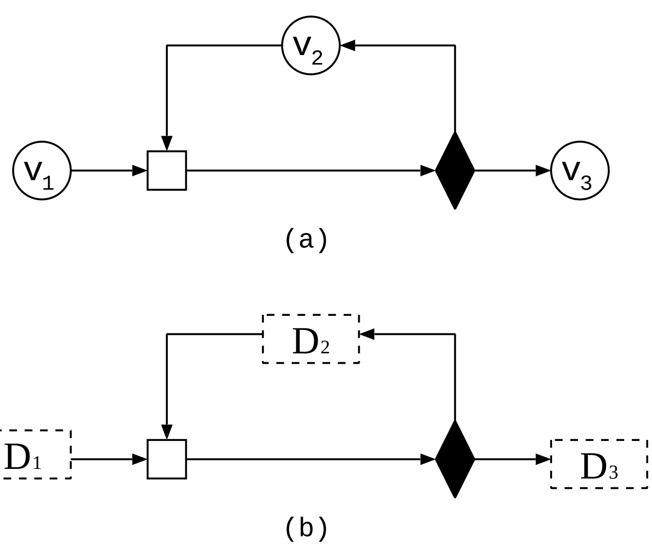


Figure 4: Construction of a loop NDSAN.

## 3 Execution time of an NDSAN

In this section we use the following terminology and notation. (See, for instance, [6, 7].) If $X$ is a random variable, then $F_X$ denotes the *probability distribution function* (*PDF*) of $X$, and $f_X$ the *probability density function* (*pdf*) of $X$. Recall that, for any $t$ in the domain of $X$, $F_X(t) = \Pr(X \leq t)$. If $X$ is a continuous variable, we have

$$F_X(t) = \int_{-\infty}^{t} f_X(x)\ dx. \tag{1}$$

Hereafter, the random variable standing for the execution time of NDSAN $D$ will be denoted by $T[D]$. This random variable can be determined as follows.

*Case 1: D is a trivial NDSAN*

Assuming that $D$ consists of the activity node $a_i$, we have $T[D] = T_i$.

*Case 2: D is not a trivial NDSAN*

By 2.1, 2.2, and 2.3 in Definition 1, $T[D]$ can be recursively determined in terms of $T[D_1], T[D_2], \dots,$ $T[D_\eta]$.

*Case 2.1: D is an acyclic NDSAN*

Consider item 2.1 in Definition 1. Let $\mathcal{P}$ be the collection of all directed paths from $source(D_0)$ to $sink(D_0)$. Let $P \in \mathcal{P}$, and write $P = v_{i_1} v_{i_2} \dots v_{i_{|P|}}$, where $|P|$ denotes the number of nodes of $P$. Let $D_{i_1}, D_{i_2}, \dots, D_{i_{|P|}}$ be the NDSANs that substitute for $v_{i_1}, v_{i_2}, \dots, v_{i_{|P|}}$. If $S_P$ is the time required for the serial execution of $D_{i_1}, D_{i_2}, \dots, D_{i_{|P|}}$, then

$$S_P = \sum_{k=1}^{|P|} T[D_{i_k}]. \tag{2}$$

(Recall that $T[D_{i_k}]$ is the random variable standing for the execution time of $D_{i_k}$, $1 \leq k \leq |P|$.)

Since the $T[D_{i_k}]$'s are independent random variables, the pdf $f_{S_P}$ of $S_P$ is given by the convolution of the pdfs $f_{T[D_{i_1}]}, f_{T[D_{i_2}]}, \dots, f_{T[D_{i_{|P|}}]}$, that is,

$$f_{S_P}(t) = (f_{T[D_{i_1}]} * f_{T[D_{i_2}]} * \dots * f_{T[D_{i_{|P|}}]})(t). \tag{3}$$

Define $f_1 = f_{T[D_{i_1}]}$ and $f_k = f_{k-1} * f_{T[D_{i_k}]}$, $2 \le k \le |P|$. Then we have, for any $t$,

$$f_k(t) = \int_0^\infty f_{k-1}(t-x) f_{T[D_{i_k}]}(x)\ dx \quad \text{and} \quad f_{S_P}(t) = f_{|P|}(t). \tag{4}$$

Following Equation (1), the PDF of $S_P$ is then given by

$$F_{S_P}(t) = \int_0^t f_{S_P}(x)\ dx. \tag{5}$$

Having described the variables $S_P$ for $P \in \mathcal{P}$, the random variable $T[D]$ is given by their maximum:

$$T[D] = \max_{P \in \mathcal{P}}\ S_P. \tag{6}$$

We remark that the variables $S_P$ are not independent, because two distinct paths in $\mathcal{P}$ may have nodes in common. Hence the PDF of $T[D]$ is given by

$$F_{T[D]}(t) = \Pr(T[D] \le t) = \Pr(S_P \le t \ \text{ for all } P \in \mathcal{P}), \tag{7}$$

but no further simplification is in general possible. To determine the pdf of $T[D]$, simply apply Equation (1):

$$f_{T[D]}(t) = (F_{T[D]})'(t). \tag{8}$$

*Case 2.2: $D$ is a decision NDSAN*

In Figure 3(b), assume that the decision node is $d_i$. Then $\alpha_i = \eta - 2$ and each node $source(D_k)$ is selected by $d_i$ with probability $p^i_k$, $k = 2, 3, \ldots, \eta - 1$. Let $X_i$ be a random variable associated with $d_i$ in such a way that

$$X_i = \begin{cases} T[D_2] \text{ with probability } p^i_2\,; \\ T[D_3] \text{ with probability } p^i_3\,; \\ \vdots \\ T[D_{\eta-1}] \text{ with probability } p^i_{\eta-1}\,. \end{cases} \tag{9}$$

Then, clearly,

$$T[D] = T[D_1] + X_i + T[D_\eta]. \tag{10}$$

In order to proceed, note that the events $X_i = T[D_k]$, $2 \le k \le \eta - 1$, are mutually disjoint, since they correspond to disjoint subdigraphs of $D$. We then have

$$f_{X_i}(t) = p^i_2\ f_{T[D_2]}(t) + p^i_3\ f_{T[D_3]}(t) + \cdots + p^i_{\eta-1}\ f_{T[D_{\eta-1}]}(t) \tag{11}$$

and

$$F_{X_i}(t) = p_2^i \; F_{T[D_2]}(t) + p_3^i \; F_{T[D_3]}(t) + \cdots + p_{\eta-1}^i \; F_{T[D_{\eta-1}]}(t). \tag{12}$$

Thus,

$$f_{T[D]}(t) = (f_{T[D_1]} * f_{X_i} * f_{T[D_\eta]})(t) \tag{13}$$

and, by Equation (1),

$$F_{T[D]}(t) = \int_0^t f_{T[D]}(x) \; dx. \tag{14}$$

*Case 2.3: D is a loop NDSAN*

In Figure 4(b), assume that the loop node is $\ell_i$. For simplicity, assume also that $\beta_i = \beta$. Recall that, at the $k$th passage through $\ell_i$, the execution flow returns to $source(D_2)$ with probability $q_k^i, k = 1, \ldots, \beta$, where $q_\beta^i = 0$ and $\beta$ is the maximum number of consecutive passages allowed through $\ell_i$.

Let $Z_k$ be the random variable standing for the total execution time of $k$ serial independent executions of $D_2$. Clearly, $Z_k$ is the sum of $k$ independent random variables, each one having distribution identical to that of $T[D_2]$. Therefore, $f_{Z_k}$ and $F_{Z_k}$ can once again be determined respectively by convolution and subsequent integration.

Consider now a random variable $Y_i$ associated with $d_i$ and such that

$$Y_i = \begin{cases} 0 & \text{with probability } 1 - q_1^i \,; \\ Z_1 & \text{with probability } q_1^i(1 - q_2^i) \,; \\ Z_2 & \text{with probability } q_1^i q_2^i (1 - q_3^i) \,; \\ & \vdots \\ Z_{\beta-1} & \text{with probability } q_1^i q_2^i \cdots q_{\beta-1}^i \,, \end{cases} \tag{15}$$

where the events $Y_i = 0,\; Y_i = Z_1,\; \ldots,\; Y_i = Z_{\beta-1}$ are all mutually disjoint. Then

$$T[D] = T[D_1] + Y_i + T[D_3], \tag{16}$$

and the functions $f_{T[D]}$ and $F_{T[D]}$ can be obtained as in Case 2.2, since the definition of $Y_i$ in Equation (15) has the same structure as that of $X_i$ in Equation (9).

# 4 Obtaining an approximate distribution of the execution time

Given an NDSAN $D$, obtaining the distribution and density functions of the target random variable $T[D]$ numerically may be an extremely costly computational task, even in simple cases. We refer the reader once again to the work by Leemis *et al.* [9], where even small networks are seen to need an elaborate mathematical analysis.

Our efforts are then directed toward seeking an approximate distribution of $T[D]$ within some required confidence level. We base our approach on collecting a random sample formed by a suitable number $N$ of independent observations of $T[D]$. Let us denote such an approximate distribution by $F^N_{T[D]}$. Once $F^N_{T[D]}$ is obtained, a frequency histogram and an approximate density $f^N_{T[D]}$ can be easily determined, as we discuss later.

First, we present a simulation algorithm that, on input $D$, outputs a single observation $t$ of the sample space of $T[D]$. Next, we deal with the question of how many times the simulation algorithm must be repeated in order to obtain $F^N_{T[D]}$ as required.

## 4.1 Simulation algorithm

The simulation algorithm is based on recursive references to subdigraphs, whose results are combined to obtain a single observation $t$ of $T[D]$. The basis of the recursion occurs when $D$ is a trivial NDSAN.

For acyclic NDSANs (refer to item 2.1 in Definition 1 and to Figure 2(b)), a single observation of $T[D]$ is obtained as follows: (i) Observations $t_1, t_2, \ldots, t_\eta$ of $T[D_1], T[D_2], \ldots, T[D_\eta]$ are obtained recursively; (ii) Denote by $C_D(t_1, t_2, \ldots, t_\eta)$ the completion time of $D$ when $T[D_i] = t_i$, $1 \leq i \leq \eta$; the determination of $C_D(t_1, t_2, \ldots, t_\eta)$ can be done by assigning weight $t_i$ to vertex $v_i$, $1 \leq i \leq \eta$, and then calculating the critical path of the resulting weighted digraph.

The description of the simulation algorithm is as follows.

Sample($D$)

1 if $D$ is a trivial NDSAN then

2 let $a_i$ be the single activity node of $D$

3 return a single observation of $T_i$

4 else if $D$ is an acyclic NDSAN then

5 let $D_1, D_2, \ldots D_\eta$ be NDSANs as in Figure 2(b)

6 return $C_D($ Sample($D_1$), Sample($D_2$), $\ldots$, Sample($D_\eta$) $)$

7 else if $D$ is a decision NDSAN then

8 let $D_1, D_2, \ldots D_\eta$ be NDSANs as in Figure 3(b)

9 let $d_i$ be the decision node of $D$

10 select $k$ from $\{2, 3, \ldots, \eta - 1\}$

11 return Sample($D_1$) $+$ Sample($D_k$) $+$ Sample($D_\eta$)

12 else if $D$ is a loop NDSAN then

13 let $D_1, D_2, D_3$ be NDSANs as in Figure 4(b)

14 let $\ell_i$ be the loop node of $D$

15 select $k$ from $\{0, 1, \ldots, \beta_i - 1\}$

16 $t_{loop} := 0$

17 repeat $k$ times

18 $t_{loop} := t_{loop} +$ Sample($D_2$)

19 return Sample($D_1$) $+$ $t_{loop}$ $+$ Sample($D_3$)

We assume that obtaining the single observation in Line 3 can be done in constant time. We also assume that the selections in Lines 10 and 15 take constant time. Note that they are related to observations of the random variables $X_i$ and $Y_i$, respectively (see Equations (9) and (15)). Then they must be made according to the probabilities expressed there. Calculating $C_D$ in Line 6 takes $O(m)$ time. (The critical path can be determined by a depth-first search starting at $source(D)$.)

Overall, the time complexity of the algorithm is determined by the maximum number of nested loop NDSANs in $D$. Suppose that $D_1, D_2, \ldots, D_\gamma$ is the longest sequence of subdigraphs of $D$ such that:

- $D_k$ is a loop NDSAN, $1 \leq k \leq \gamma$;
- $D_{k+1}$ is a proper subdigraph of $D_k$, $1 \leq k \leq \gamma - 1$.

Let $\bar{\beta} = \max\{\beta_i \mid 1 \leq i \leq n_\ell\}$. Then in each $D_k$ at most $\bar{\beta} - 1$ consecutive iterations are performed. Hence, the worst-case time complexity of the algorithm is $O(\bar{\beta}^{\gamma} m)$. Although $\gamma = O(n)$ and $\bar{\beta}$ can be arbitrarily large, for most typical NDSANs the values of $\gamma$ and $\bar{\beta}$ are bounded by small constants. Thus the algorithm has, in practice, an $O(m)$ time complexity.

### 4.2 Repeated executions of the simulation algorithm

Since $F_{T[D]}$ is a continuous variable, we may resort to the same statistic on $F^{N}_{T[D]}$ as the Kolmogorov-Smirnov (KS) test. We refer the reader to [7] (Section 13.5) and to [8] (Section 3.3.1) for more details on what follows.

Let $t_1, t_2, \ldots, t_N$ be a random sample of $T[D]$, obtained by $N$ independent executions of the simulation algorithm. Define $F^{N}_{T[D]}$ as

$$F^{N}_{T[D]}(x) = \frac{\mid \{t_i \mid t_i \leq x\} \mid}{N}. \tag{17}$$

The KS test is based on the difference between $F_{T[D]}(x)$ and $F^{N}_{T[D]}(x)$. To measure this difference, we form the statistic

$$K_N = \sup_{x \geq 0} \mid F^{N}_{T[D]}(x) - F_{T[D]}(x) \mid \tag{18}$$

(hereafter referred to as the KS statistic), which may be visualized as the maximum distance (error), along the ordinate axis, between the plots of $F_{T[D]}(x)$ and $F^{N}_{T[D]}(x)$ over the range of all possible $x$ values. It can be shown (see [7], p. 346) that the distribution of $K_N$ does not depend on $F_{T[D]}$. As a consequence, $K_N$ can be used as a nonparametric random variable for constructing a confidence band for $F_{T[D]}$.

Let $K^{\varepsilon}_{N}$ denote a value satisfying the relation

$$\Pr(K_N \leq K^{\varepsilon}_{N}) = 1 - \varepsilon \tag{19}$$

for some $0 < \varepsilon < 1$. Following Equations (18) and (19), we have:

$$\begin{aligned} 1 - \varepsilon &= \Pr(\sup_{x \geq 0} \mid F^{N}_{T[D]}(x) - F_{T[D]}(x) \mid \leq K^{\varepsilon}_{N}) \\ &= \Pr( \mid F^{N}_{T[D]}(x) - F_{T[D]}(x) \mid \leq K^{\varepsilon}_{N} \text{ for all } x \geq 0) \\ &= \Pr(F^{N}_{T[D]}(x) - K^{\varepsilon}_{N} \leq F_{T[D]}(x) \leq F^{N}_{T[D]}(x) + K^{\varepsilon}_{N} \text{ for all } x \geq 0). \end{aligned} \tag{20}$$

The last equality in Equation (20) shows that the functions $F^{N}_{T[D]}(x) - K^{\varepsilon}_{N}$ and $F^{N}_{T[D]}(x) + K^{\varepsilon}_{N}$ yield a confidence band, with confidence level $1 - \varepsilon$, for the unknown distribution function $F_{T[D]}(x)$.

Table 1: Some critical values $K_N^\varepsilon$ for $K_N$.

| $N$ | $\varepsilon = 0.20$ | $\varepsilon = 0.10$ | $\varepsilon = 0.05$ | $\varepsilon = 0.01$ |
|---|---|---|---|---|
| 10 | 0.32 | 0.37 | 0.41 | 0.49 |
| 20 | 0.23 | 0.26 | 0.29 | 0.36 |
| 30 | 0.19 | 0.22 | 0.24 | 0.29 |
| 40 | 0.17 | 0.19 | 0.21 | 0.25 |
| 50 | 0.15 | 0.17 | 0.19 | 0.23 |
| large | $1.07/\sqrt{N}$ | $1.22/\sqrt{N}$ | $1.36/\sqrt{N}$ | $1.63/\sqrt{N}$ |

Some of the values $K_N^\varepsilon$ of the distribution of $K_N$ are given in Table 1 (see [7], p. 411). From Table 1 we have, for example, $K_{50}^{0.20} = 0.15$. Thus

$$\Pr(K_{50} \leq K_{50}^{0.20}) = \Pr(K_{50} \leq 0.15) = 1 - 0.20 = 0.80. \tag{21}$$

That is, by repeating the simulation algorithm $N = 50$ times, the probability that the error $K_N$ is at most 0.15 is 0.80. More accurate results can be obtained by using the last row of Table 1. For example, by requiring a maximum error 0.02 with confidence 95%, we have $\varepsilon = 0.05$ and

$$\Pr(K_N \leq K_N^{0.05}) = \Pr(K_N \leq 0.02) = 0.95. \tag{22}$$

For large $N$, Table 1 gives us $K_N^{0.05} = 1.36/\sqrt{N}$. From $1.36/\sqrt{N} = 0.02$ we conclude that $N = 4624$ repeated executions of the simulation algorithm are needed in this case.

We can summarize the application of the KS statistic as follows.

1. Stipulate the maximum error $e$ and the confidence level $c$.
2. Set $\varepsilon = 1 - c$ and determine from Table 1 the value of $N$ for which $K_N^\varepsilon \approx e$.
3. Run the simulation algorithm $N$ times and obtain a random sample $t_1, t_2, \ldots, t_N$.
4. Let $F_{T[D]}^N$ be as in Equation (17).
5. If needed, an approximate density $f_{T[D]}^N$ can be determined as follows, assuming $t_1 \leq t_2 \leq \cdots \leq t_N$. For some step value $\delta > 0$, let

$$f_{T[D]}^N(t_{1+k\delta}) = \frac{F_{T[D]}^N(t_{1+k\delta}) - F_{T[D]}^N(t_{1+(k-1)\delta})}{t_{1+k\delta} - t_{1+(k-1)\delta}}, \quad k = 1, 2, \ldots, \lfloor N/\delta \rfloor - 1. \tag{23}$$

   For instance, for $\delta = 25$ we compute the values

$$f_{T[D]}^N(t_{26}) = \frac{F_{T[D]}^N(t_{26}) - F_{T[D]}^N(t_1)}{t_{26} - t_1}\ ,\ f_{T[D]}^N(t_{51}) = \frac{F_{T[D]}^N(t_{51}) - F_{T[D]}^N(t_{26})}{t_{51} - t_{26}}\ ,$$

and so on. (We remark that better, nonparametric methods are available, as explained in [14], for example.)

# 5 Computational experiments

## 5.1 A typical development process

Figure 5 shows a simple, yet typical, development process represented by a NDSAN $D$ with $S_a = \{a_1, \ldots, a_{27}\}$, $S_b = \{b_1, \ldots, b_8\}$, $S_d = \{d_1\}$, and $S_\ell = \{\ell_1, \ldots, \ell_7\}$.

Table 2 describes the activity nodes, whose durations are expressed in days. Here, all $T_i$'s follow *triangular densities*, which are suitable for describing single activities of a business or industrial process [5]. The pdf $f_X$ of a triangular variable $X$ with parameters $x_1 < x_2 < x_3$ is given by:

$$f_X(x) = \begin{cases} 0, & x < x_1; \\ \frac{y_0}{x_2 - x_1}(x - x_1), & x_1 \leq x < x_2; \\ \frac{y_0}{x_3 - x_2}(x_3 - x), & x_2 \leq x < x_3; \\ 0, & x \geq x_3, \end{cases} \tag{24}$$

where $y_0 = \frac{2}{x_3 - x_1}$. Table 3 shows the probabilities associated with the decision node $d_1$, Table 4 those associated with the loop nodes $\ell_1$ through $\ell_7$.

If we require a maximum error of 2% with confidence 95%, the KS statistic yields $K_N^{0.05} = 1.36/\sqrt{N}$ (see Table 1). From $1.36/\sqrt{N} = 0.02$, we conclude that $N = 4624$ repeated executions of $\mathsf{Sample}(D)$ are required. Each of these executions can be represented by a tree of recursive calls, as follows. Let $D_i$ be the trivial NDSAN consisting of the activity node $a_i$, $1 \leq i \leq 27$, and, for $i < j$, let $D_{i,j}$ be the NDSAN defined as the maximal connected induced subdigraph $D'$ of $D$ satisfying $source(D') = a_i$ and $sink(D') = a_j$. Figure 6 depicts the tree of recursive calls. For example, $D_{5,27}$ is a decision NDSAN, and in order to obtain a single observation of $T[D_{5,27}]$ we first recursively obtain observations of $T[D_5], T[D_6], T[D_{7,26}]$, and $T[D_{27}]$.

The frequency histogram of the resulting sample of $T[D]$ is shown in Figure 7 for 1-wide bins. Each bin is an interval of the form $(a, b]$ and abscissae in the figure give the values of $b$. The histogram suggests that $f_{T[D]}$ follows a bimodal pattern. Figure 7 also shows the fitting curve

$$f_1(x) = 2115 \text{ lognorm}(2.379610, 0.125138, x) + 2509 \text{ lognorm}(3.853650, 0.072067, x), \tag{25}$$

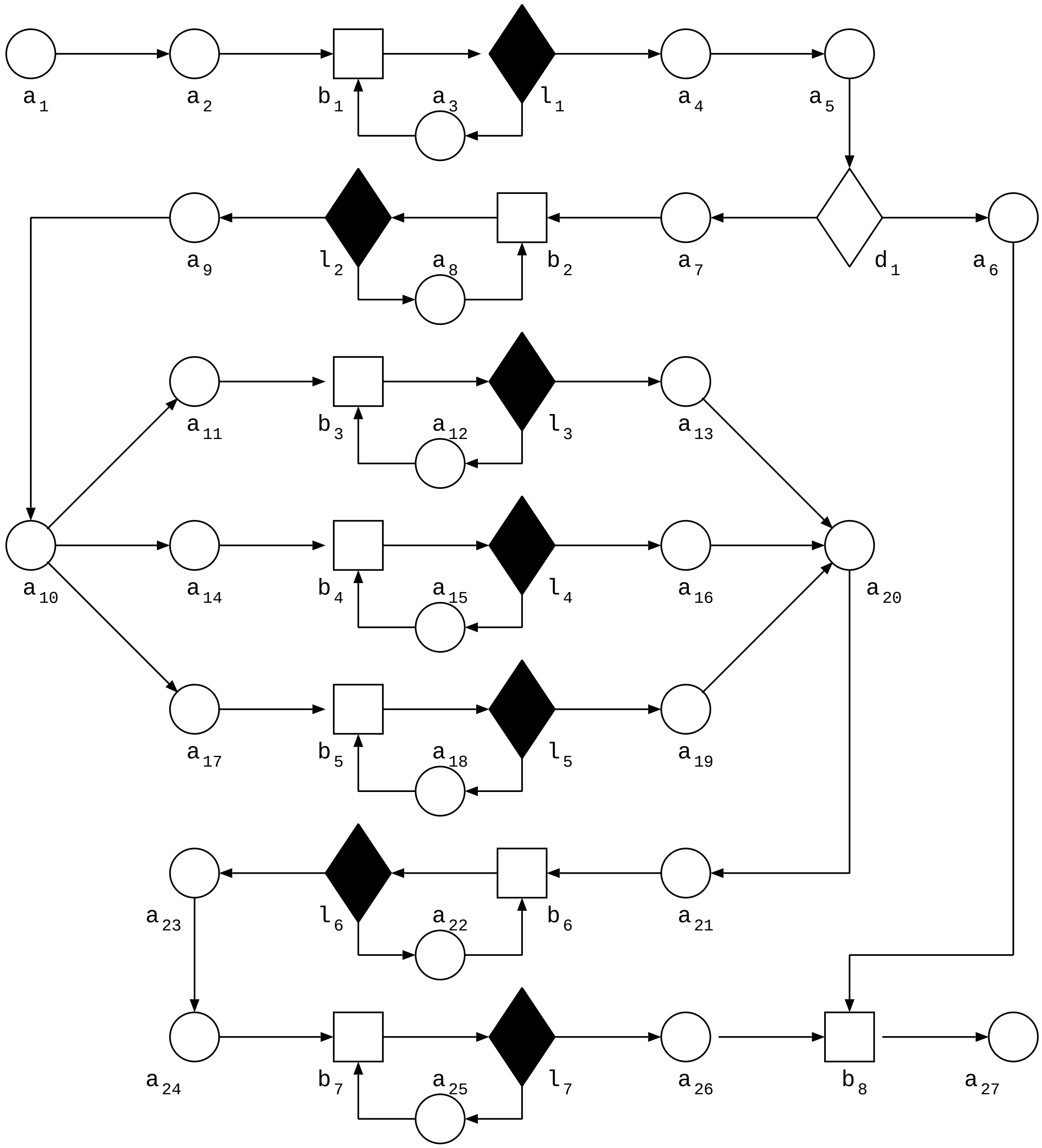


Figure 5: An NDSAN representing a development process.

Table 2: Activity nodes of the NDSAN of Figure 5.

| Node | Description | Density parameters |
|---|---|---|
| $a_1$ | requirement analysis | 2, 4, 5 |
| $a_2$ | contract negotiation | 1, 2.5, 3.5 |
| $a_3$ | renegotiation | 1, 1.5, 2 |
| $a_4$ | contract conclusion | 0.5, 1, 1.5 |
| $a_5$ | contract presentation | 0.5, 1, 1.5 |
| $a_6$ | project abandonment | 0.5, 1, 1.5 |
| $a_7$ | system analysis | 4, 8, 12 |
| $a_8$ | system analysis refinement | 0.5, 2, 3 |
| $a_9$ | system analysis conclusion | 0.5, 1, 1.5 |
| $a_{10}$ | division into modules | 0.5, 1, 1.5 |
| $a_{11}$ | 1st module implementation | 4, 6, 12 |
| $a_{12}$ | 1st module refinement | 1, 2, 3 |
| $a_{13}$ | 1st module conclusion | 0.5, 1, 1.5 |
| $a_{14}$ | 2nd module implementation | 4, 6, 12 |
| $a_{15}$ | 2nd module refinement | 1, 2, 3 |
| $a_{16}$ | 2nd module conclusion | 0.5, 1, 1.5 |
| $a_{17}$ | 3rd module implementation | 4, 6, 12 |
| $a_{18}$ | 3rd module refinement | 1, 2, 3 |
| $a_{19}$ | 3rd module conclusion | 0.5, 1, 1.5 |
| $a_{20}$ | module integration | 0.5, 1.5, 3 |
| $a_{21}$ | integration test | 1, 3.5, 4 |
| $a_{22}$ | error fixing | 0.5, 1, 1.5 |
| $a_{23}$ | product deployment | 0.5, 1, 1.5 |
| $a_{24}$ | client test | 2, 4, 6 |
| $a_{25}$ | error fixing | 0.5, 1, 1.5 |
| $a_{26}$ | production dispatch | 0.5, 1, 1.5 |
| $a_{27}$ | project documentation | 0.5, 1, 1.5 |

Table 3: Probabilities associated with the decision node $d_1$ in Figure 5.

| Node | Description | Outcome | Next activity | Probability |
|---|---|---|---|---|
| $d_1$ | contract accepted? | `yes` | $a_7$ | 55% |
| | | `no` | $a_6$ | 45% |

Table 4: Probabilities associated with the loop nodes in Figure 5.

| Node | Description | Outcome | Next activity | 1st iter. | 2nd iter. | 3rd iter. |
|---|---|---|---|---|---|---|
| $\ell_1$ | negotiation finished? | `yes` | $a_4$ | 50% | 80% | 100% |
| | | `no` | $a_3$ | 50% | 20% | 0% |
| $\ell_2$ | use cases approved? | `yes` | $a_9$ | 10% | 50% | 100% |
| | | `no` | $a_8$ | 90% | 50% | 0% |
| $\ell_3$ | 1st module passed? | `yes` | $a_{13}$ | 20% | 50% | 100% |
| | | `no` | $a_{12}$ | 80% | 50% | 0% |
| $\ell_4$ | 2nd module passed? | `yes` | $a_{16}$ | 20% | 50% | 100% |
| | | `no` | $a_{15}$ | 80% | 50% | 0% |
| $\ell_5$ | 3rd module passed? | `yes` | $a_{19}$ | 20% | 50% | 100% |
| | | `no` | $a_{18}$ | 80% | 50% | 0% |
| $\ell_6$ | integration passed? | `yes` | $a_{23}$ | 60% | 80% | 100% |
| | | `no` | $a_{22}$ | 40% | 20% | 0% |
| $\ell_7$ | client test passed? | `yes` | $a_{26}$ | 20% | 50% | 100% |
| | | `no` | $a_{25}$ | 80% | 50% | 0% |

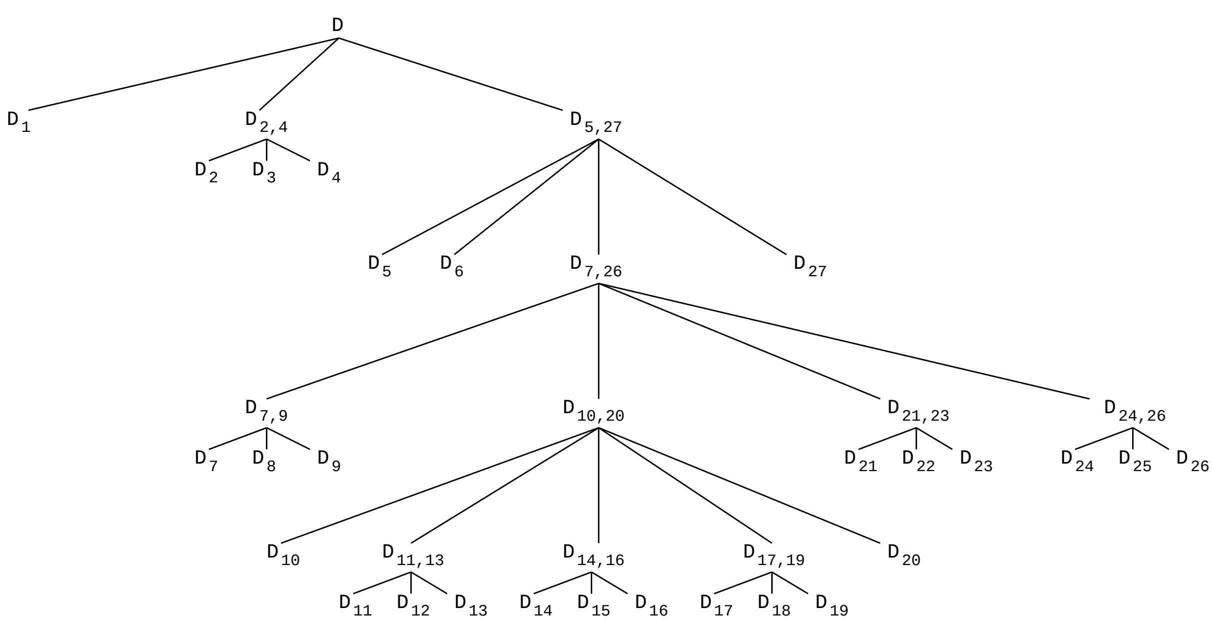


Figure 6: Recursive calls invoked by $\mathsf{Sample}(D)$; $D$ is the NDSAN of Figure 5.

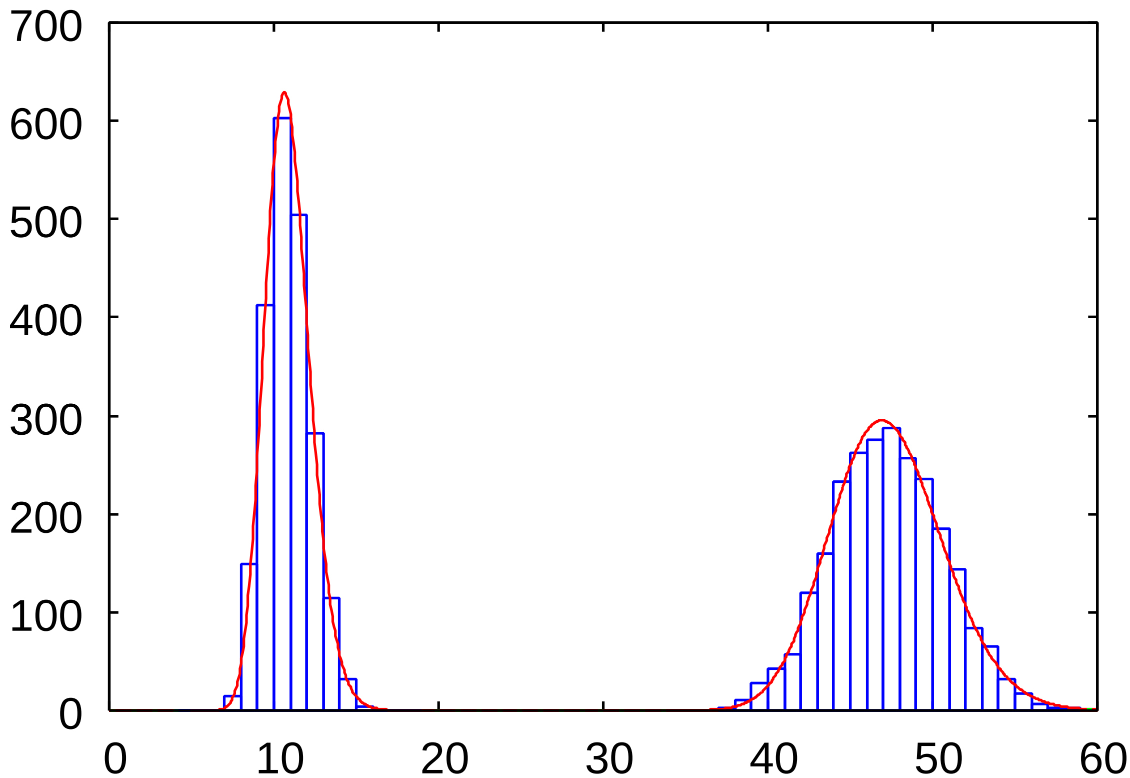


Figure 7: Fitting curve drawn on the frequency histogram of $T[D]$ for $N = 4624$ and 1-wide bins; $D$ is the NDSAN of Figure 5.

where $\text{lognorm}(\mu, \sigma, x)$ is the density function of the *log normal distribution* [15] with parameters $\mu$ (the *scale* parameter) and $\sigma$ (the *shape* parameter):

$$\text{lognorm}(\mu, \sigma, x) = \frac{1}{\sigma\sqrt{2\pi}x} e^{-(\ln x - \mu)^2/2\sigma^2}. \tag{26}$$

The function $f_1(x)$ is therefore proportional to the sum of two densities, the former yielding positive values over the range $(7, 16]$, the latter over $(37, 60]$.

The approximate $F^N_{T[D]}$ and $f^N_{T[D]}$ are shown in Figures 8 and 9 respectively, the latter with $\delta = 25$ in Equation (23).

### 5.2 A paper reviewing process

Figure 10 shows an NDSAN $D$ representing the typical peer-review process of scientific publishing. Table 5 describes the activity nodes, whose durations are once again expressed in days. The $T_i$'s follow *truncated normal distributions*. In the third column of Table 5, each line shows a pair $\mu_i, \sigma_i^2$, standing for the mean and the variance of $T_i$, respectively. Each $T_i$ is restricted to lie in the range

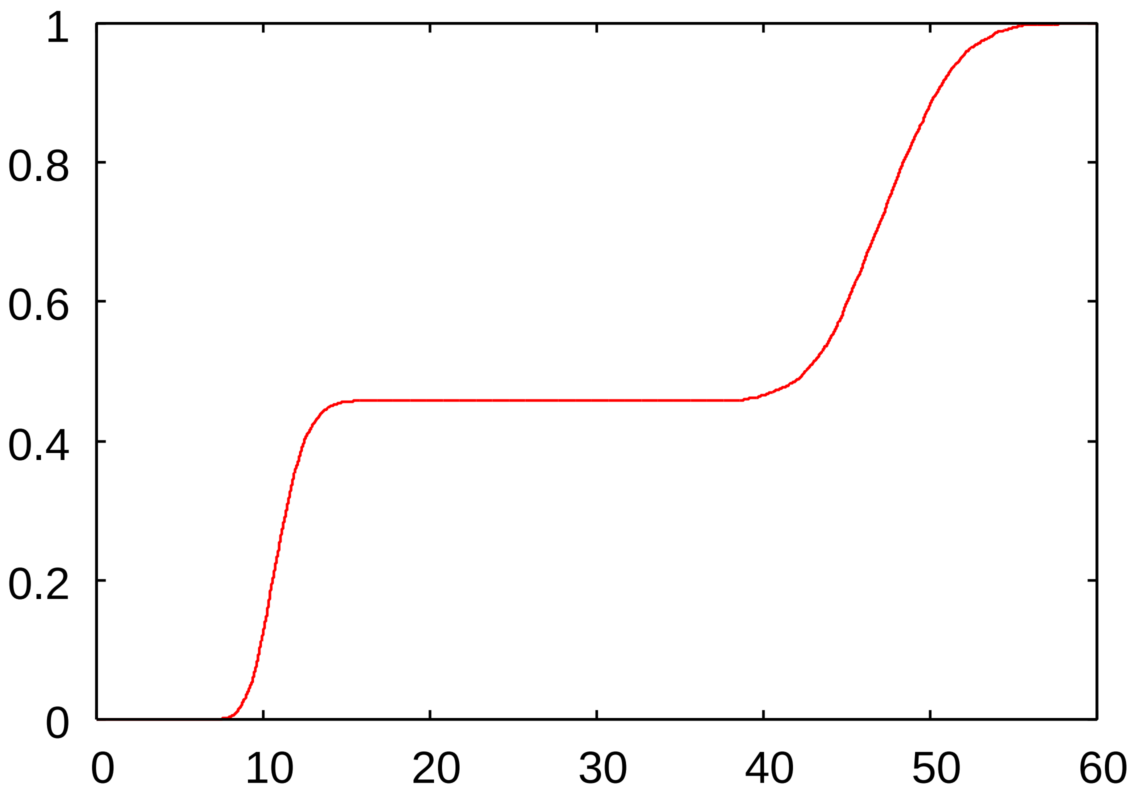


Figure 8: Approximate distribution $F^N_{T[D]}$ for $N = 4624$; $D$ is the NDSAN of Figure 5.

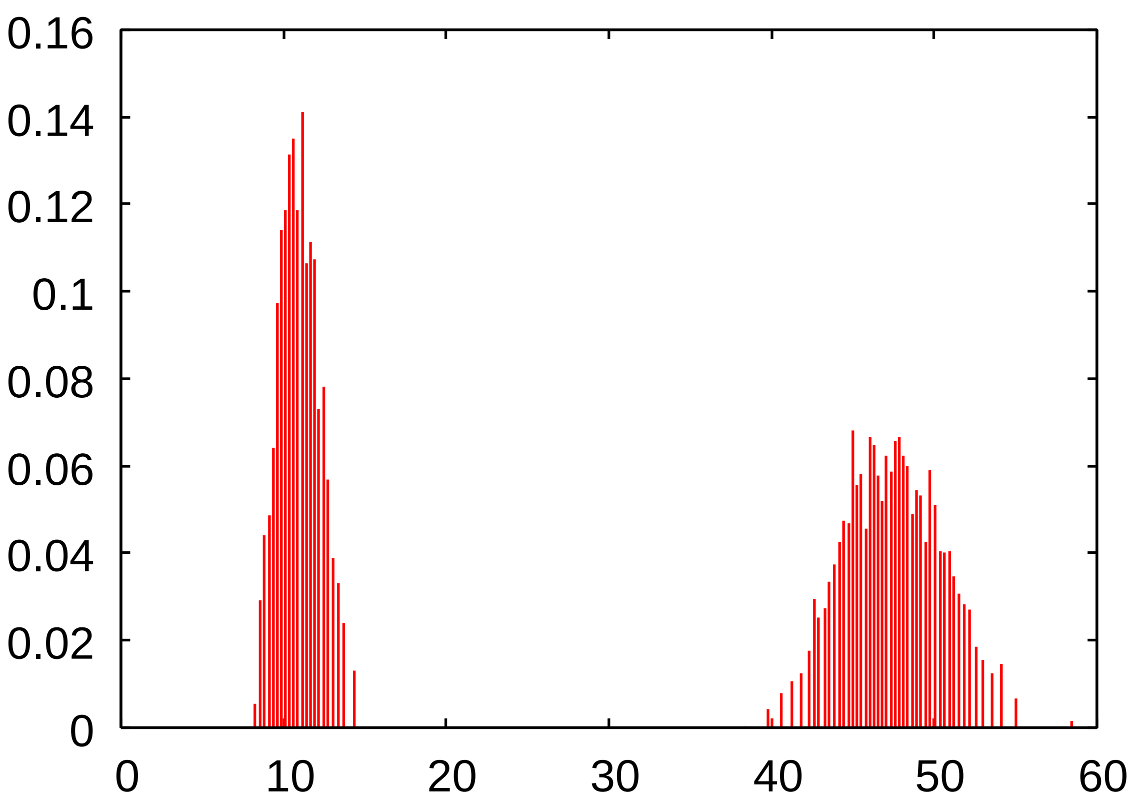


Figure 9: Approximate density $f^N_{T[D]}$ for $N = 4624$ and $\delta = 25$; $D$ is the NDSAN of Figure 5.

$[\mu_i - 3\sigma_i, \mu_i + 3\sigma_i]$. Table 6 shows the probabilities associated with the decision node $d_1$, Table 7 the probabilities associated with the loop nodes $\ell_1$ and $\ell_2$.

For the same 2% error and 95% confidence as above, we give the results from $N = 4624$ repeated executions of $\mathsf{Sample}(D)$ in Figures 11 through 13. These figures show, respectively, the fitting curve $f_2(x) = 4624 \text{ lognorm}(4.965323, 0.421285, x)$ drawn on the frequency histogram of $T[D]$ for 1-wide bins, the approximate distribution of $T[D]$, and the approximate density of $T[D]$ (with $\delta = 25$ in Equation (23)).

# 6 Ongoing work

The introduction of the constraint that each activity node requires certain amounts of finitely available resources to execute gives raise to the so-called *activity networks with constrained resources.* The problem associated with such networks is known as RCPSP (Resource-Constrained Project Scheduling Problem) [2]. The RCPSP has many variations, but even the deterministic RCPSP with fixed activity durations is NP-hard [1].

Resource-Constrained NDSANs (RCNDSANs) combine stochastic activity durations, nondeterminism, and constrained resources. We are currently targeting the simulation algorithm of RCNDSANs, based on iterating the combination of two phases as many times as necessary for accuracy. The first phase is responsible for obtaining a non-stochastic, deterministic instance of the input RCNDSAN, by selecting one of its possible execution paths. (Here, the term "path" stands for a plausible non-stochastic, deterministic scenario: a network represented by a directed acyclic graph with fixed topology and fixed activity durations.) The second phase consists of employing a heuristic procedure for the solution of the deterministic RCPSP. The repeated execution of "path selection" combined with "scheduling heuristics" will generate close approximations to the probability distribution of the variables under analysis.

We remark that our simulation algorithms turn out to be low-cost tools for the identification of the factors that most strongly influence completion time. After a simulation round, if needed, changes in the structure of the NDSAN/RCNDSAN under analysis can be proposed in order to improve its performance. Several simulation rounds may be rapidly performed until the desired efficiency is actually achieved.

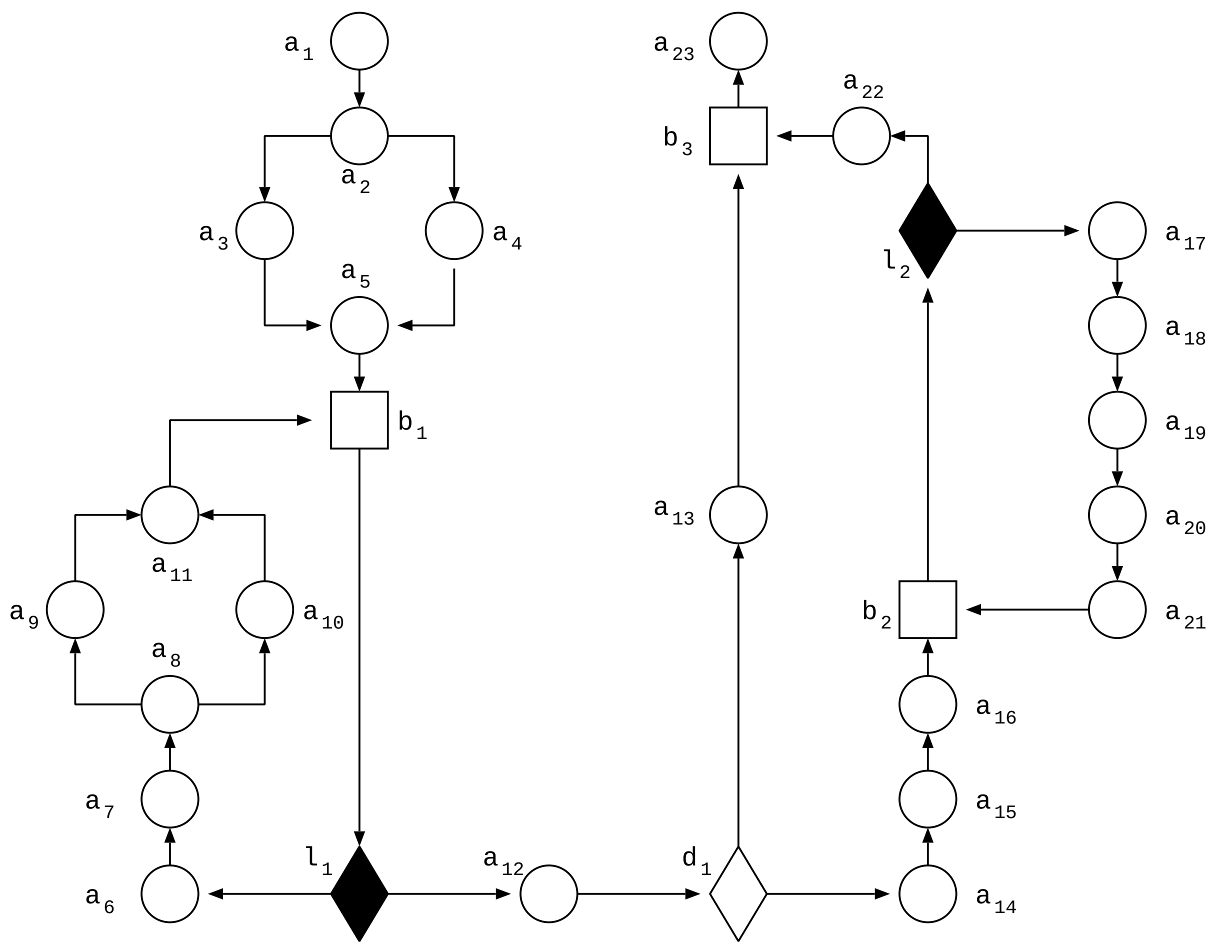


Figure 10: An NDSAN representing a paper reviewing process.

Table 5: Activity nodes of the NDSAN in Figure 10.

| Node | Description | Mean, variance |
|---|---|---|
| $a_1$ | authors submit paper | 1, 0.1 |
| $a_2$ | editor sends paper to referees 1 and 2 | 1, 0.1 |
| $a_3$ | referee 1 processes the paper | 90, 45 |
| $a_4$ | referee 2 processes the paper | 90, 45 |
| $a_5$ | editor processes reports | 2, 0.2 |
| $a_6$ | editor sends reports to authors | 1, 0.1 |
| $a_7$ | authors perform modifications | 14, 7 |
| $a_8$ | editor sends revised version to referees 1 and 2 | 1, 0.1 |
| $a_9$ | referee 1 processes revised version | 14, 7 |
| $a_{10}$ | referee 2 processes revised version | 14, 7 |
| $a_{11}$ | editor processes new reports | 2, 0.2 |
| $a_{12}$ | editor checks agreement of reports | 1, 0.1 |
| $a_{13}$ | editor makes final decision based on two reports | 2, 0.2 |
| $a_{14}$ | editor sends paper to referee 3 | 1, 0.1 |
| $a_{15}$ | referee 3 processes the paper | 90, 45 |
| $a_{16}$ | editor processes report of referee 3 | 2, 0.2 |
| $a_{17}$ | editor sends report of referee 3 to authors | 1, 0.1 |
| $a_{18}$ | authors perform modifications | 14, 7 |
| $a_{19}$ | editor sends revised version to referee 3 | 1, 0.1 |
| $a_{20}$ | referee 3 processes revised version | 14, 7 |
| $a_{21}$ | editor processes new report of referee 3 | 2, 0.2 |
| $a_{22}$ | editor makes final decision based on three reports | 2, 0.2 |
| $a_{23}$ | editor sends final result to authors | 1, 0.1 |

Table 6: Probabilities associated with the decision node $d_1$ in Figure 10.

| Node | Description | Outcome | Next activity | Probability |
|---|---|---|---|---|
| $d_1$ | referees agree? | `yes` | $a_{13}$ | 75% |
| | | `no` | $a_{14}$ | 25% |

Table 7: Probabilities associated with the loop nodes in Figure 10.

| Node | Description | Outcome | Next activity | 1st iter. | 2nd iter. | 3rd iter. |
|---|---|---|---|---|---|---|
| $\ell_1$ | no need of modifications? | `yes` | $a_{12}$ | 81% | 98% | 100% |
| | | `no` | $a_6$ | 19% | 2% | 0% |
| $\ell_2$ | no need of modifications? | `yes` | $a_{22}$ | 90% | 99% | 100% |
| | | `no` | $a_{17}$ | 10% | 1% | 0% |

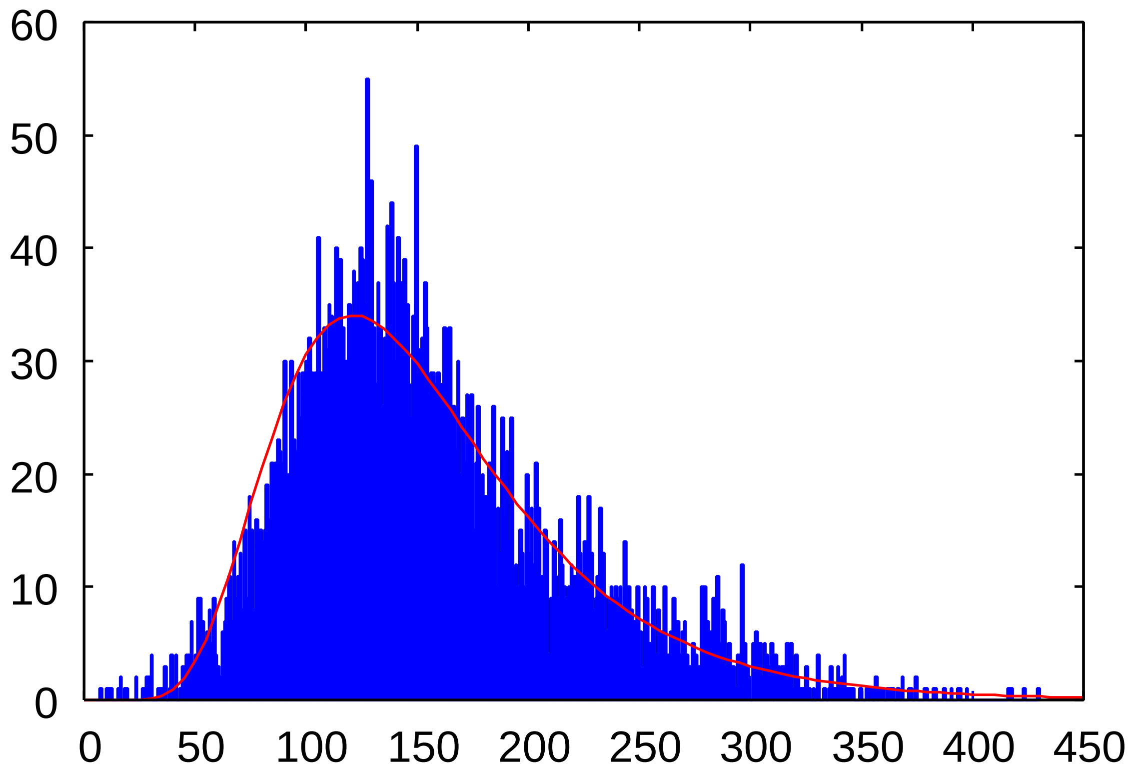


Figure 11: Fitting curve drawn on the frequency histogram of $T[D]$ for $N = 4624$ and 1-wide bins; $D$ is the NDSAN of Figure 10.

## References

[1] J. Blazewicz, J. K. Lenstra, and A. H. G. Rinnooy Kan. Scheduling subject to resource constraints: classification and complexity. *Discrete Applied Mathematics* 5 (1983) 11–24.

[2] E. Demeulemeester, W. Herroelen, and B. De Reyck. A classification scheme for project scheduling. In J. Weglarz (ed.), *Project Scheduling: Recent Models, Algorithms and Applications*, Kluwer, Dordrecht, 1999, pp. 1–26.

[3] E. Demeulemeester, W. Herroelen, and M. Vanhoucke. Discrete time/cost trade-offs in project scheduling with time-switch constraints. *Journal of the Operational Research Society* 53,7 (2002) 741–751.

[4] S. E. Elmaghraby. *Activity Networks*, Wiley, New York, 1977.

[5] M. Evans, N. Hastings, and B. Peacock. *Statistical Distributions*, 3rd ed., Wiley, New York, 2000, pp. 187–188.

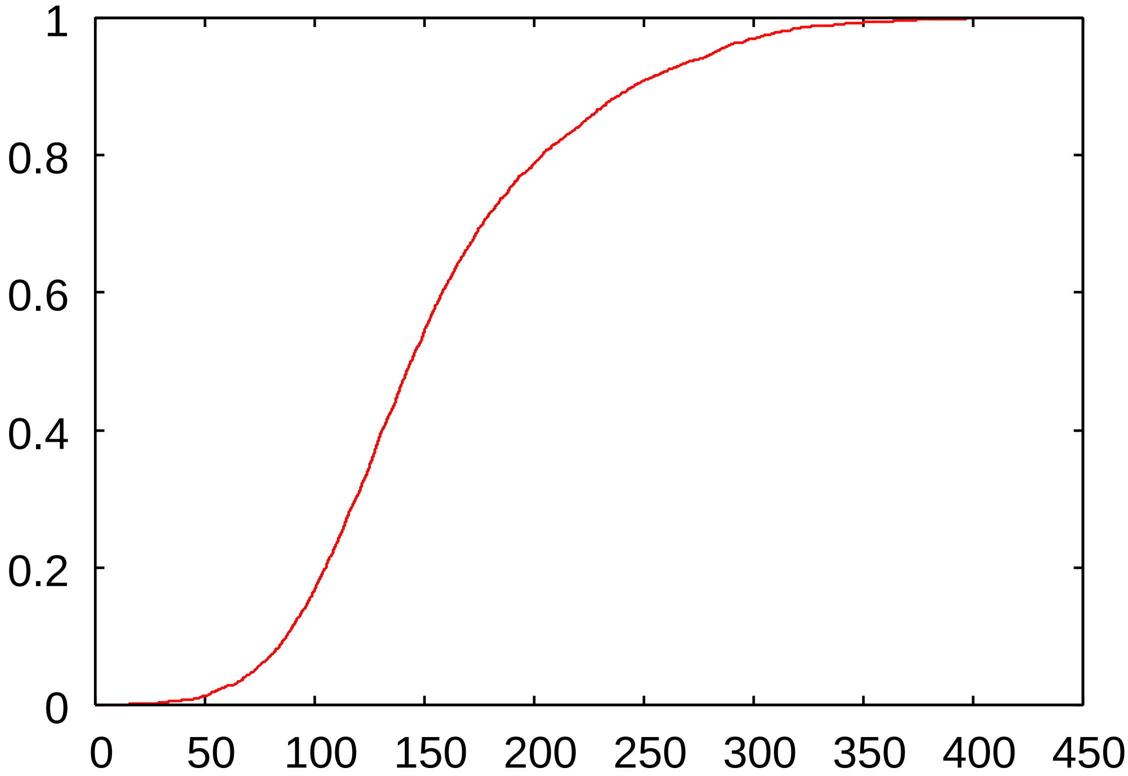


Figure 12: Approximate distribution $F^N_{T[D]}$ for $N = 4624$; $D$ is the NDSAN of Figure 10.

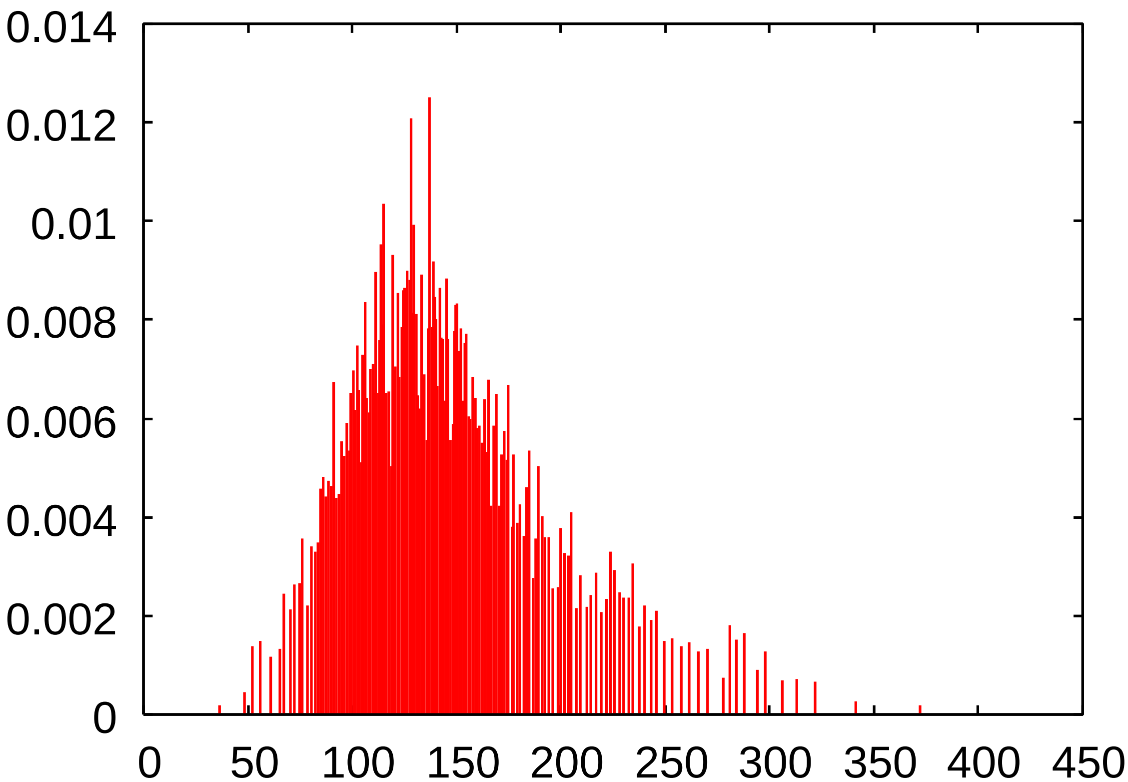


Figure 13: Approximate density $f^N_{T[D]}$ for $N = 4624$ and $\delta = 25$; $D$ is the NDSAN of Figure 10.

[6] W. Feller. *An Introduction to Probability Theory and Its Applications*, Vol. I, 2nd ed., Wiley, New York, 1991.

[7] P. G. Hoel. *Introduction to Mathematical Statistics*, 3rd ed., Wiley, New York, 1965.

[8] D. E. Knuth. *The Art of Computer Programming*, Vol. II, 3rd ed., Addison-Wesley, Reading, Massachusetts, 1997.

[9] L. M. Leemis, M. J. Duggan, J. H. Drew, J. A. Mallozzi, and K. W. Connel. Algorithms to calculate the distribution of the longest path length of a stochastic activity network with continuous activity durations. *Networks* 48 (2006) 143–165.

[10] J. J. Moder, C. R. Phillips, and E. W. Davis. *Project Management with CPM, PERT and Precedence Diagramming.* Van Nostrand Reinhold Company, New York, 3rd ed., 1983.

[11] L. J. Moore and E. R. Clayton. *GERT Modeling and Simulation: Fundamentals and Applications*. Petrocelli/Charter, New York, 1976.

[12] R. A. V. Olaguibel and J. M. T. Goerlich. Heuristic algorithms for resource-constrained project scheduling. In R. Slowinski and J. Werglarz (eds.), *Advances in Project Scheduling*, Elsevier, Amsterdam, 1989, pp. 113–134.

[13] D. R. Shier. *Network Reliability and Algebraic Structures.* Oxford University Press, New York, 1991.

[14] B. W. Silverman. *Density Estimation for Statistics and Data Analysis.* Chapman & Hall, London, 1986.

[15] E. W. Weisstein. Log normal distribution. Wolfram MathWorld, http://mathworld.wolfram.com/LogNormalDistribution.html.